SafeRoute: Enhancing Traffic Scene Understanding via a Unified Deep Learning and Multimodal LLM

Supplementary Material

S1. Additional Results

This document provides supplementary experimental results and analysis to complement our main paper.

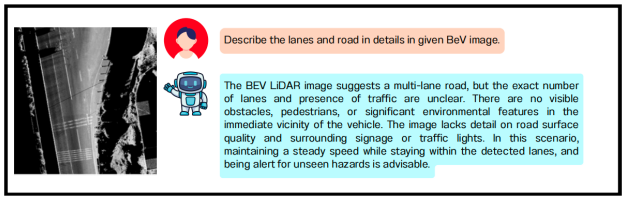

Figure S1. Initial Evaluation of MiniGPT-v2 on BEV LiDAR Image Without Task-Specific Training

S1.1. Initial Evaluation of MiniGPT-v2 on BEV LiDAR Image

An initial evaluation of the MiniGPT-v2 model, as illustrated in Figure S1, assessed its capability to describe lanes and road features from a BEV LiDAR image without task-specific training. The model's response, "The BEV LiDAR image suggests a multi-lane road, but the exact number of lanes and presence of traffic are unclear," reveals several limitations. Firstly, it lacks precise lane and road element descriptions, failing to specify the exact number or type of lanes, which is crucial for autonomous vehicle navigation. Secondly, key road features such as intersections, lane boundaries, and road markings remain undetected, limiting the model's spatial and contextual awareness. Lastly, the response offers generalized observations with minimal task-specific relevance, providing no insights on lane conditions, road quality, or potential hazards—critical elements for HD map updates and autonomous driving decisions. These findings highlight the need for domain-specific fine-tuning to enhance the model's ability to interpret and reason about road environments with greater accuracy.

S1.2. VLM Response on BEV Image with Unannotated Data

In the initial approach to training a Vision-Language Model (VLM) for road element detection, we employed unannotated Bird's Eye View (BEV) LiDAR images with structured metadata, aiming for autonomous identification of lane markings, lane types, and road attributes. However, as illustrated in Figure S4, the model exhibited critical limitations in distinguishing lane types and accurately interpreting road elements. When prompted, the model identified

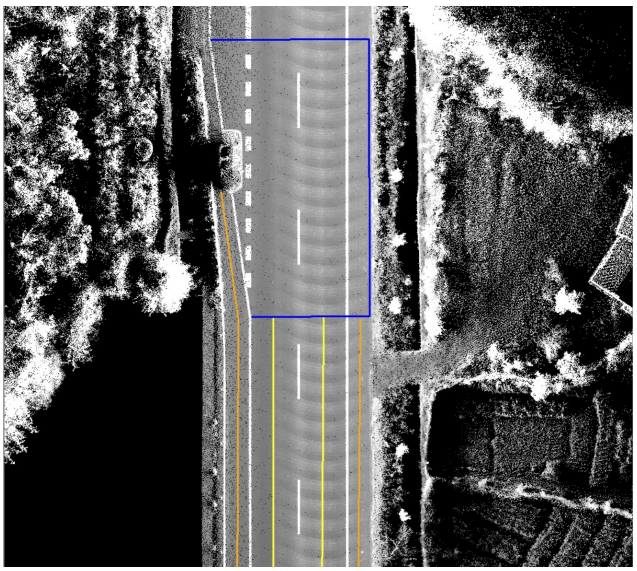

Figure S2. Annotate figure

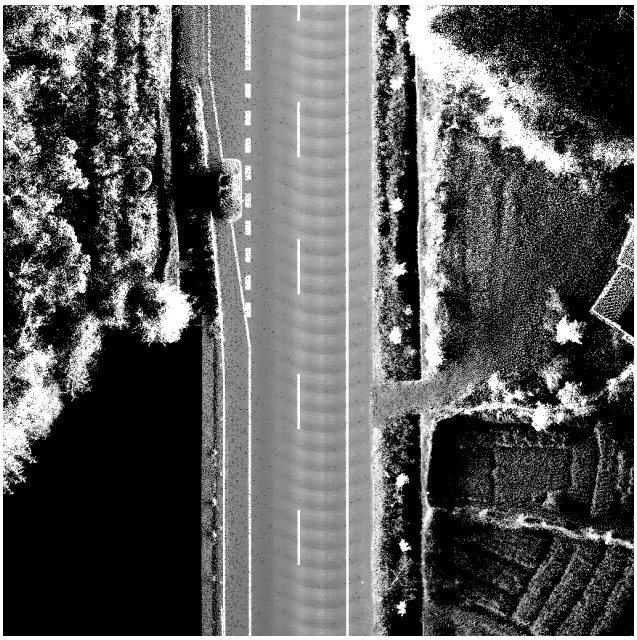

Figure S3. Unannotate figure

only two lanes, failing to recognize the presence of bicycle lanes and incorrectly reporting "No bicycle lanes visible," despite their existence in the image. This misclassification suggests a lack of fine-grained lane differentiation, likely

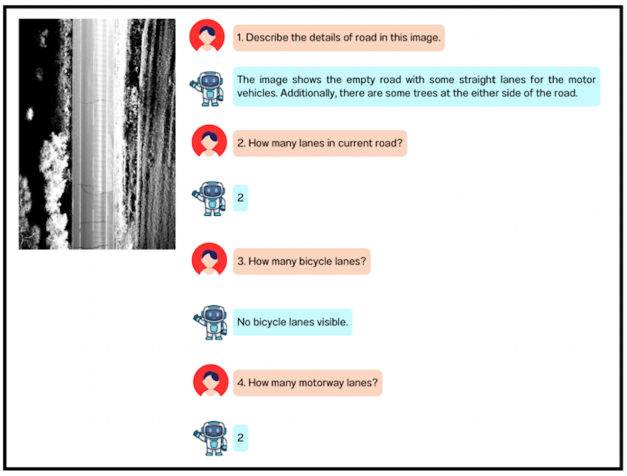

Figure S4. Example of VLM Response on BEV Image when trained with Unannotated Data

due to the absence of task-specific annotations. Furthermore, while the model correctly identified two motorway lanes, its response appeared to rely on generalized assumptions rather than precise understanding, merging bicycle and motorway lanes into a simplified interpretation. These findings highlight the need for enhanced training strategies, including explicit annotations and pre-processing, to improve the VLM's ability to accurately detect and categorize diverse lane types for autonomous driving applications and HD map updates.

S1.3. Task-Specific Trained MiniGPT-v2 Model

After fine-tuning the VLM model with task-specific polylines coded datasets and incorporating guided captioning, its ability to understand and describe road elements has significantly improved, as illustrated in Figure S5. Initially, the model provided vague and generalized responses, but post-training, it now accurately identifies key road components, including motor vehicle and bicycle lanes, and clearly describes their structure and arrangement. It correctly quantifies lane counts, distinguishing between two motorway lanes and two bicycle lanes, demonstrating a refined capacity for lane differentiation. Additionally, the model accurately recognizes the absence of intersections or lane change zones, assesses the quality of the point cloud data as "very clear," and categorizes the road scene as a "normal clear city road." These improvements highlight the effectiveness of specialized training in enhancing the model's spatial reasoning and structured comprehension of road environments, making it more suitable for autonomous driving applications and HD mapping.

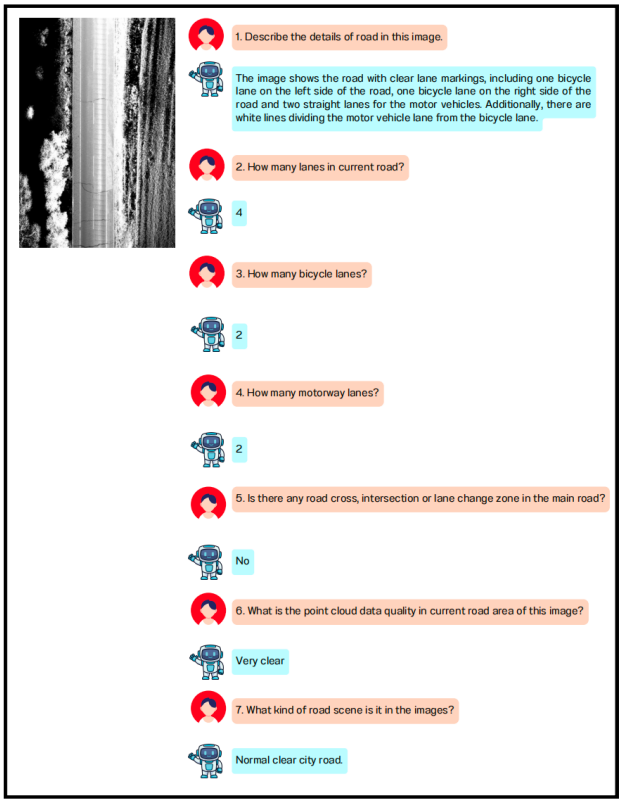

Figure S5. Task-Specific Trained MiniGPT-v2 Model for Lane Elements Understanding

S1.4. Rural Road Scene Understanding

Furthermore, the model has demonstrated an improved ability to generalize and interpret road environments, as illustrated in Figure S6. When analyzing an underdeveloped rural road scene, it provided a clear and detailed description of both the surroundings and key road features. The model correctly identified the road type as an "underdeveloped rural road" and accurately noted the presence of "green vegetation and some trees on either side of the road." This highlights its capability to move beyond lane detection and contextualize broader environmental elements. Additionally, the model successfully identified the absence of road crossovers, intersections, or lane change zones, further enhancing its structural understanding of rural road scenes. These findings underscore the model's capacity for comprehensive road perception, making it more adaptable to diverse driving conditions in both urban and rural environments.

S1.5. Lane Lines in Rural Environment

As illustrated in Figure S7, the model effectively identifies lane lines and weather conditions in a rural road setting, demonstrating its multimodal reasoning capabilities. When

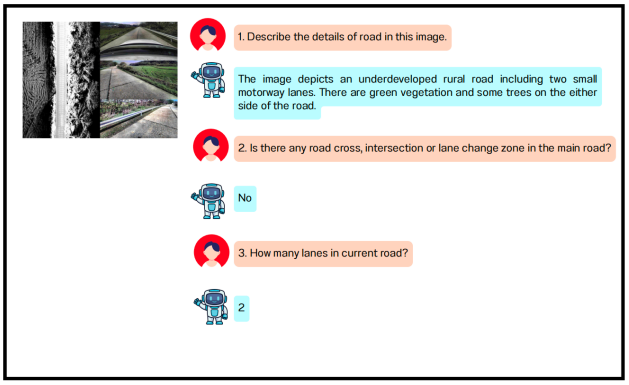

Figure S6. Rural Road Scene Understanding

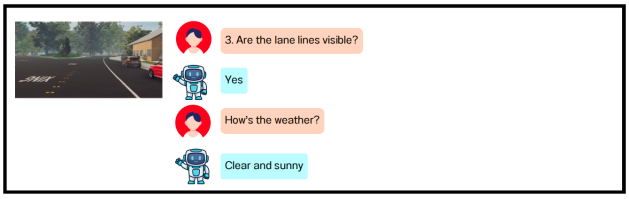

Figure S7. Lane Lines Visible in a Rural Environment with Clear Weather

prompted, it correctly confirmed the visibility of lane lines and accurately described the weather as "clear and sunny." This result highlights the model's ability to interpret road elements not only in urban environments but also in rural settings, where varying conditions can impact driving decisions. The ability to integrate environmental context, such as weather conditions, into its reasoning is particularly crucial for autonomous navigation, as adverse weather can significantly influence lane visibility and overall driving safety. The model's success in this scenario underscores its potential for enhancing perception systems in diverse real-world driving conditions.

S1.6. Suboptimal Result: Incorrect Lane Count

Lastly, as illustrated in Figure S8, a suboptimal case highlights a minor limitation in the Vision-Language Model (VLM)'s ability to accurately identify the total number of lanes. The model incorrectly detected four lanes instead of five, likely due to worn-off lane markings and unclear LiDAR data points, which impaired its ability to distinguish all lanes correctly. This case underscores the model's challenges in processing visually degraded inputs and the need for further refinements to enhance its robustness under imperfect conditions. However, despite this minor error, the model demonstrated strong performance in road symbol recognition, correctly identifying and interpreting directional arrows for one-way traffic and lane change indications. Additionally, it provided a comprehensive scene

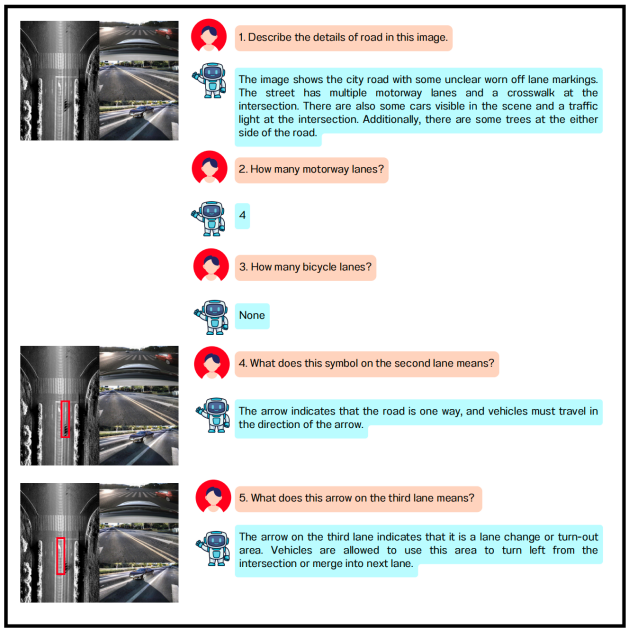

Figure S8. Suboptimal Result: VLM Model Inaccurately Estimates Lane Count

description, accurately noting the presence of vehicles, a crosswalk, trees, and a traffic light at the intersection. These results highlight the model's strong generalization capabilities in understanding complex road environments, even as occasional inaccuracies in lane detection indicate areas for future optimization.

S1.7. Suboptimal Result: Incorrect Lane Detection

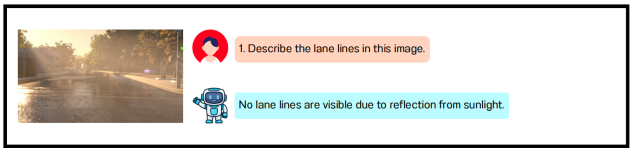

Figure S9. Suboptimal Case: Incorrect Lane Line Detection due to Misinterpretation of Sunlight Reflection

As illustrated in Figure S9, the model encounters challenges in accurately identifying lane lines under complex lighting conditions. When prompted, it responded, "No lane lines are visible due to reflection from sunlight," despite the fact that lane lines are partially visible in the image. This misinterpretation highlights a limitation in the model's ability to differentiate between environmental factors affecting lane visibility, such as reflections from sunlight on wet surfaces and the residual effects of rainfall. While the model attempts to reason about the cause of reduced visibility, its explanation is incomplete, failing to account for the interplay of multiple overlapping factors. This scenario under-

scores the need for further training and refinement, particularly in improving the model's robustness against challenging visual conditions and enhancing its capacity to provide more precise lane detection explanations in adverse environments.

S1.8. Specific Task Prompts

During training, we created several task-specific prompts to guide the model in understanding specific features of the map data. These task prompts helped focus the model on key elements and enhanced its interpretive abilities.

The road elements understanding task-specific prompts are listed below:

1. Describe the lanes and road in detail in this image: The model is prompted to provide a short description summarizing the scene and key road elements.
   Example: "The image shows a normal city road with clearly visible lanes and an intersection. There are both bicycle and motorway lanes present."
2. What is the object marked? This prompt guides the model to identify a highlighted object or feature, such as a lane or intersection.
   Example: "The marked object is a motorway lane."
3. What does this arrow mean? For situations where directional cues, such as arrows, are marked in the image.
   Example: "The arrow indicates a lane change direction for merging into the motorway."
4. How many bicycle lanes are there? The model is prompted to count the number of bicycle lanes in the scene.
   Example: "There is one bicycle lane in this image."
5. How many motorway lanes are there? Similar to the above, this prompt focuses on identifying motorway lanes.
   Example: "There are two motorway lanes present."
6. Is there any cross-section or intersection? The model is asked to identify and describe intersections or lane-change zones.
   Example: "Yes, there is a lane-change zone at the intersection."

The adaptive lane detection and reasoning task-specific prompts are:

1. Describe the lane lines.
   Example: "The lane lines are clearly visible and well-marked along the road."
2. Are the lane lines visible?
   Example: "Yes, the lane lines are visible and distinct in the image."
3. If not, what is the reason for their invisibility?
   Example: "The lane lines are not visible due to heavy rainfall."
4. Why are the lane lines not visible (e.g., due to heavy rain, degradation, or darkness)?
   Example: "The lane lines are partially visible due to degradation.

S1.9. Evaluation Metrics

To comprehensively evaluate the performance of our multimodal Vision-Language Model (VLM) on the MAPLM-QA benchmark, we define four key evaluation categories also known as Fine-grained QA:

- **SCN (Scene Understanding):** Determines the type of road scene present in the image.
- **QLT (Point Cloud Quality Analysis):** Assesses the quality of the point cloud data in the given frame.
- **LAN (Lane Counting):** Evaluates the correctness of the predicted number of lanes in the current road scene.
- **INT (Intersection Recognition):** Identifies whether a road cross, intersection, or lane-change zone exists.

For holistic performance assessment, we introduce two aggregated metrics: Frame-Overall Accuracy (FRM) and Question-Overall Accuracy (QNS).

S1.9.1. Frame-Overall Accuracy (FRM)

FRM measures whether all lane and road element-related responses are correctly answered within a single frame. It is defined as:

$$FRM = \frac{1}{N}\sum_{k=1}^{N}(SCN_k \cdot QLT_k \cdot LAN_k \cdot INT_k) \quad (5)$$

where $SCN_k, QLT_k, LAN_k, INT_k \in \{0, 1\}$ are binary indicators representing the correctness of each response type for frame k, and N denotes the total number of frames in the test set. A frame is considered fully correct ($FRM = 1$) if all four components are correctly identified.

S1.9.2. Question-Overall Accuracy (QNS)

QNS measures the overall correctness ratio across all categories, providing a frame-wise mean accuracy across the four evaluation types. It is computed as:

$$QNS = \frac{1}{N}\sum_{k=1}^{N}\frac{SCN_k + QLT_k + LAN_k + INT_k}{4} \quad (6)$$

where the numerator represents the number of correctly answered sub-tasks per frame, and the denominator ensures equal weight across all evaluation categories.

These evaluation metrics enable a rigorous comparison of different VQA models on the MAPLM-QA benchmark, providing insights into both fine-grained road scene understanding and frame-level holistic accuracy. By leveraging FRM and QNS, we effectively quantify the interpretability, consistency, and reliability of the model's responses, making them critical benchmarks for assessing multimodal VLMs in autonomous driving perception.

Table S1. Comparison of our MLLM with State-of-the-Art MLLMs on MAPLM-QA Dataset.

Method	Backbone	Additional Learning	Modalities		Metrics (↑)					
			Img	PC	LAN	INT	QLT	SCN	QNS	FRM
LLaVA	Vicuna-7B	P+IT+LoRA	✓	✓	75.4	77.53	82.4	95.53	82.72	52.27
MAPLM	Vicuna-7B	P+IT	✓	✓	72.93	78.4	82.27	94.93	82.13	50.53
MAPLM	Vicuna-7B	P+IT+LoRA	✓	✓	78.53	83.2	84.33	96.00	85.52	57.99
Ours	LLAMA-2-7B	IT+LoRA	×	✓	75.80	77.53	82.33	95.67	82.83	53.87

Metrics: LAN (Lane Counting), INT (Intersection Recognition), QLT (Point Cloud Quality), SCN (Scene Understanding), FRM (Frame-Overall Accuracy: 1 if all LAN, INT, QLT, and SCN are correct; else 0), QNS (Question-Overall Accuracy). Training paradigms: P (Pretraining), IT (Instruction Tuning), LoRA (Low-Rank Adaptation), PC (Point Cloud).

S1.10. Our MLLM's Competitive Performance

The results in the Table 6 demonstrate that our LLAMA-2-7B-based MLLM, despite being more light weight and computationally efficient, achieves competitive performance compared to Vicuna-7B-based models, making it a strong contender in multimodal road scene understanding.

Our model achieves 82.83% QNS and 53.87% FRM, ranking second-best among all models, closely trailing behind MAPLM (P+IT+LoRA, Vicuna-7B), which achieves the highest 85.52% QNS and 57.99% FRM. Notably, our model outperforms MAPLM (P+IT, Vicuna-7B), which only reaches 82.13% QNS and 50.53% FRM, demonstrating that our instruction-tuned model with LoRA achieves better generalization than standard pretraining-based approaches. Furthermore, our model surpasses LLaVA (P+IT+LoRA, Vicuna-7B) in QNS (82.83% vs. 82.72%) and FRM (53.87% vs. 52.27%), proving its strong reasoning capabilities while requiring fewer computational resources.

Although MAPLM (P+IT+LoRA, Vicuna-7B) achieves the highest accuracy in LAN (78.53%), INT (83.2%), and QLT (84.33%), our model remains highly competitive, attaining 75.80% LAN, 77.53% INT, and 82.33% QLT, making it a viable alternative for scenarios where computational efficiency is a priority. Unlike LLaVA and MAPLM models, which require both image and point cloud inputs, our model relies solely on point clouds, yet still delivers comparable or superior results. This highlights the effectiveness of our instruction tuning and LoRA-based adaptation strategy, proving that our model is highly efficient while maintaining state-of-the-art multimodal reasoning performance.

S1.11. Dataset Composition

We used a total of 10,000 annotated Bird's-Eye View (BEV) images for training and 2,000 raw images for testing, sourced from the MAPLM dataset. Each image was annotated with polylines representing lanes and cross-sections, as detailed in Algorithm 3.3 of the main paper. The training dataset included a variety of road environments, ranging from urban streets to highways and rural roads, ensuring that the model could generalize well across different conditions. The annotations were consistently applied to all images to maintain accuracy and ensure the Vision-Language Model (VLM) could reliably learn these patterns during training.

This preprocessing technique proved critical in enhancing the model's ability to understand and reason about road elements, leading to improved lane detection accuracy and overall scene understanding. The process laid the foundation for training a robust multimodal model capable of accurately detecting and reasoning about road features in a variety of complex environments. The combination of structured geometric data with visually encoded annotations using polylines is a key innovation in this work, enabling the VLM to effectively interpret road structures.

S1.12. Specific Instruction Prompting for Road Elements Understanding

To implement task-guided instruction prompting, we transformed raw map data from the MAPLM dataset into a format readable by the LLaMA-2 language model, as described in the main paper. This step was essential for ensuring that the model could handle the specific task prompts effectively, guiding it to focus on relevant road features like lanes, intersections, and other map elements. The raw map data from the MAPLM dataset typically includes geometric coordinates and attributes for lanes, cross-sections, and other road features in a general JSON-like format. This format is not easily interpretable by an LLM designed to work with natural language inputs.

We converted this data into a natural language description following the COCO Caption Description Format, which LLaMA-2 can process, as shown in Figures 3.9 and 3.10 of the main paper. For instance, a raw MAPLM dataset entry (see Figure 3.8 in the main paper was transformed into a structured caption such as: "The scene features a normal city road with very clear data quality. It includes four lanes: a bicycle lane from [379.14, 1024.0] to [353.74, 351.14], a motorway from [438.73, 1024.0] to [517.88,

Raw MAPLM dataset entry

```
Global Attributes:
• Valid: true
• Data Quality: Very clear
• Scene Type: Normal city road
Lanes:
1. Type: Bicycle lane
 Geometry: [379.14, 1024.0] → [381.09, 560.2] →
 [353.74, 351.14]
2. Type: Motorway
 Geometry: [438.73, 1024.0] → [438.73, 555.32]
Cross Section:
• Type: Lane change zone
 Geometry: [336.43, 103.98], [594.75, 102.74],
     [595.76, 553.7], [403.63, 554.05]
```

Figure S10. Raw MAPLM dataset entry about lanes. This example shows geometric data for lanes (bicycle lane, motorway) and a cross-section (lane-change zone) with coordinates.

Converted raw data into COCO caption style

```
"image_id": "FR1"
"caption": "The scene features a normal city road
    with very clear data quality. It includes four
    lanes: a bicycle lane from [379.14, 1024.0] to
    [353.74, 351.14], a motorway from [438.73, 1024.0]
    to [438.73, 555.32], and a motorway from
    [517.88, 1024.0] to [520.6, 554.55], along with a
    lane-change zone at the intersection with vertices
    [336.43, 103.98], [594.75, 102.74],
    [595.76, 553.7] and [403.63, 554.05]."
```

Figure S11. Raw data conversion into a format that an LLM can process. The MAPLM dataset entry is transformed into a COCO-style caption, enabling LLaMA-2 to interpret road features.

1024.0] and a motorway from [520.6, 554.55], along with a lane-change zone at the intersection with vertices [336.43, 103.981], [594.75, 102.741], [595.76, 553.71] and [403.63, 554.05].” This structured annotation methodology enhances the dataset’s utility for evaluating lane detection and road scene understanding across both standard and challenging environmental conditions.

S2. Supplementary Notes: Source Code Availability

All source code and datasets will be made publicly available on GitHub. Due to the large size of the dataset, it cannot be uploaded here or shared via online cloud platforms, in adherence to the blind review policy. The code and related resources will be released on GitHub or an official website, which will be announced soon.

Algorithm 3.1 Drawing Color-Coded Polylines on Raw Images.

```
Input: List of images with map data (lane geometries, attributes, and cross-sections)
Output: Annotated images with color-coded polylines for road features
Algorithm: Draw Color-Coded Polylines From Map Data
    # Step 1: Loop through each image in the dataset
    foreach image in dataset do
        # Step 2: Extract lane geometries and attributes
        lanes ← image["lanes"]
        cross_sections ← image.get("cross", None)
        # Step 3: Initialize an empty image canvas
        canvas ← createCanvas(image["width"], image["height"])
        # Step 4: Loop through each lane and draw polylines based on attributes
        foreach lane in lanes do
            # Step 4.1: Extract lane geometry and lane type
            lane_type ← lane["attr"]
            points ← lane["geom"]
            # Step 4.2: Set polyline color based on lane type
            if lane_type == "motorway" then
                color ← Yellow
            else if lane_type == "bicycle lane" then
                color ← Orange
            # Step 4.3: Draw polyline for the lane
            drawPolyline(canvas, points, color)
        # Step 5: Draw cross-sections (if available)
        if cross_sections exists then
            # Step 5.1: Extract cross-section geometry
            cross_points ← cross_sections[0]["geom"]
            # Step 5.2: Set polyline color for cross-sections
            cross_color ← Blue
            # Step 5.3: Draw polyline for cross-sections
            drawPolyline(canvas, cross_points, cross_color)
        # Step 6: Save or display the annotated image
        saveImage(canvas, "annotated_" + image["image_id"])
```

Figure S12. Algorithm for drawing color-coded polylines on raw images. The process extracts lane geometries, assigns colors based on lane types (e.g., Yellow for motorways, Orange for bicycle lanes, Blue for cross-sections), and annotates the image accordingly.